

**Ethereum HD Wallet in JS**

Exercise



[www.kingslanduniversity.com](http://www.kingslanduniversity.com/) [updated 18 February 2018]

In this exercise, you shall implement a hierarchical wallet (​**HD wallet​**) for Ethereum​ **​**in JavaScript using the​[Ethers.js library](https://github.com/ethers-io/ethers.js)​. The wallet will hold a ​**BIP39 mnemonic phrase​** and will allow ​**deriving private keys​** and addresses from it.

# Introduction

This exercise will focus on the concept of hierarchical wallets (HD Wallets), based on ​**BIP39​** and ​**BIP44​** specs. The​**[HDNode](https://github.com/ethers-io/ethers.js/blob/master/wallet/hdnode.js)​** [class](https://github.com/ethers-io/ethers.js/blob/master/wallet/hdnode.js)​in the ​**Ethers.js​** library holds a ​**seed key​** + the ability to ​**derive private keys​** by a certain derivation path. Let’s play with it.

**Create​** a new JavaScript project (call it ​*ethereum-hd-wallet.js​*). Initialize a package.json file using:

**npm init**

To get ​**ethers.js,​** install it through the NPM package manager:

## npm install --save ethers

### 1. Play with BIP39 and BIP44 Online

First, play a bit with the ​**BIP39 online implementation​** here:

<https://iancoleman.io/bip39>​. Generate/load mnemonics, derive Ethereum and Bitcoin keys and addresses.

**2. Restore HD Wallet by Existing Mnemonic**

Restore an HD node by given existing mnemonic words.

## upset fuel enhance depart portion hope core animal innocent will athlete snack

A ​**Hierarchical Deterministic Wallet​** represents a large tree of private keys, which can be reproduced from an initial seed. Each node in the tree is represented by an ​**HDNode​** which can be branched.

To create an HD node from a mnemonic, use **ethers.utils.HDNode.fromMnemonic​**:

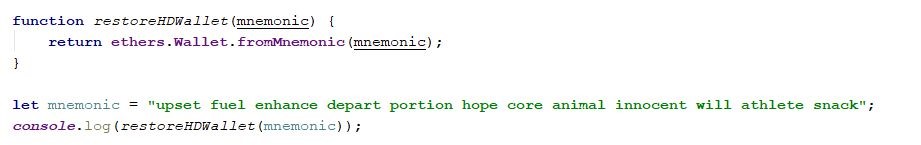
|  |
| --- |
| const​ ethers = ​require​(​'ethers'​);  function​ ​restoreHDNode​(mnemonic) {  ​return​ ethers.utils.HDNode.fromMnemonic(mnemonic);  } let​ mnemonic = ​"upset fuel enhance depart portion hope core animal innocent will athlete snack" console​.log(restoreHDNode(mnemonic)); |

The HD node stores a ​**key pair ​**(publicKeyBytes has been truncated):

|  |
| --- |
| HDNode {  ​keyPair​:  KeyPair { ​privateKey​:  ​'0x507e63d76f1ac1da6257643a8820f3b1c3e3805adb88022ec49628e2f96a2bae'​, publicKey​:  '0x047018b3b6d7a946d838e15fd3723a847401a263b13ec40d5b3cd8e4a4c6a20af282a3fd402d e2a9713d836547377c489b55acd0c8da3d8d6b7844c90932d95667'​, compressedPublicKey​:  '0x037018b3b6d7a946d838e15fd3723a847401a263b13ec40d5b3cd8e4a4c6a20af2'​, ​publicKeyBytes​:  [ ​3​,  ​112​,  ​10​, ​242​ ] }, ​privateKey​:  ​'0x507e63d76f1ac1da6257643a8820f3b1c3e3805adb88022ec49628e2f96a2bae'​, ​publicKey​:  ​'0x037018b3b6d7a946d838e15fd3723a847401a263b13ec40d5b3cd8e4a4c6a20af2'​, |
| ​address​: ​'0xF44db43680983B4705CA1eA2Ec15fD4412e3738F'​, ​chainCode​:  ​'0x117381d1bdc3f788ec0da2157cc4b63a029450786738621c7c1aa6563e9402dd'​,  ​index​: ​0​,  ​depth​: ​0​, ​mnemonic​:  ​'upset fuel enhance depart portion hope core animal innocent will athlete snack'​,  ​path​: ​'m'​ } |

The ​**depth​** represents the sequence of the derivation path or hierarchy of this node.

When an ​**ethers.Wallet ​**instance is created from a mnemonic, it actually uses **HDNode.fromMnemonic​**, ​**derives once​**, and from the new HD node, it takes the private key to build the wallet.



Console output:

|  |
| --- |
| Wallet {  ​signingKey​:  SigningKey { mnemonic​:  ​'upset fuel enhance depart portion hope core animal innocent will athlete snack'​,  ​path​: ​"m/2147483692'/2147483708'/2147483648'/0/0"​, ​privateKey​:  ​'0xf0858a4efb32ae920092378f82ddefda93e10d1bdb5b4a538887836a8dc80cf3'​, ​keyPair​:  KeyPair { ​privateKey​:  ​'0xf0858a4efb32ae920092378f82ddefda93e10d1bdb5b4a538887836a8dc80cf3'​, ​publicKey​:  '0x047983de57876c66c537153e2185031c2d1e8d59bbb0268eddca0273655c2357385871cbbd6a08 |
| 40df5023499159ee909b63b168c5882a6ef48d966c2b691466c9'​, ​compressedPublicKey​:  ​'0x037983de57876c66c537153e2185031c2d1e8d59bbb0268eddca0273655c235738'​, ​publicKeyBytes​:  [ ​3​,  ​87​,  ​56​ ] }, ​publicKey​:  '0x047983de57876c66c537153e2185031c2d1e8d59bbb0268eddca0273655c2357385871cbbd6a08  40df5023499159ee909b63b168c5882a6ef48d966c2b691466c9'​,  ​address​: ​'0xfDd85780CB96f4712a869aB4d04f9D80c3CeE283'​ },  ​provider​: ​undefined​ } |

### 3. New Random HD Wallet

Generate a new random HD node (generate random mnemonics)

To create a random HD node, you can either do

**ethers.Wallet.createRandom​** and build the HD node from the mnemonic, or you can build the mnemonic from 16 bytes of entropy using **ethers.utils.HDNode.entropyToMnemonic**

|  |
| --- |
| function​ ​generateMnemonic​() {  ​let​ randomEntropyBytes = ethers.utils.randomBytes(​16​);  ​return​ ethers.utils.HDNode.entropyToMnemonic(randomEntropyBytes);  } function​ ​generateRandomHDNode​() {  ​let​ mnemonic = generateMnemonic();  ​return​ ethers.utils.HDNode.fromMnemonic(mnemonic);  }  console​.log(generateRandomHDNode()); |

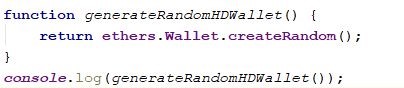
Console output:

HDNode {

​keyPair​:

|  |
| --- |
| KeyPair { ​privateKey​:  ​'0x372d52b1db050ffc3102c1872ef2162806834b9ba198e31ffb436411909ace37'​, ​publicKey​:  '0x0435ec730f69b26d8619cc03f72511a09b2f28779649d06764b7cdad67ba2f8a566b4cf6fd7369 1480771f9f9fb6a367a649cbe1e1dace02c280c392755f8f7657'​, ​compressedPublicKey​:  '0x0335ec730f69b26d8619cc03f72511a09b2f28779649d06764b7cdad67ba2f8a56'​, ​publicKeyBytes​:  [ ​3​,  ​53​,  ​138​,  ​86​ ] }, ​privateKey​:  ​'0x372d52b1db050ffc3102c1872ef2162806834b9ba198e31ffb436411909ace37'​, ​publicKey​:  ​'0x0335ec730f69b26d8619cc03f72511a09b2f28779649d06764b7cdad67ba2f8a56'​, ​address​: ​'0x637312D72fD37481f4ee10F8620F40C56977b8d4'​, ​chainCode​:  ​'0xaf147ce7e208dad526c4e0e22f11b772dcf6f9233b93eac97ce2da9bc13b0ff1'​,  ​index​: ​0​,  ​depth​: ​0​, ​mnemonic​:  ​'basic flag nerve slide fox auction liquid goddess concert master guess strategy'​,  ​path​: ​'m'​ } |

You can also create a random HD wallet using ​**ethers.Wallet.createRandom**



Console output:

Wallet {

​signingKey​:

SigningKey {

|  |
| --- |
| ​mnemonic​:  ​'buzz clay occur achieve army battle okay rotate dash meadow detail safe'​, ​path​: ​"m/2147483692'/2147483708'/2147483648'/0/0"​, ​privateKey​:  ​'0x24a472d7d04b232add85a563ed8599270958f478506f44d2d8a931f64e827889'​, ​keyPair​:  KeyPair { ​privateKey​:  ​'0x24a472d7d04b232add85a563ed8599270958f478506f44d2d8a931f64e827889'​, ​publicKey​:  '0x042a46d3f965d8e0e99785765a9875046712f193b311323f8937a77210074465953bc0b704a60c 75e3ae70fa47468aea873e35dd44a915486c8bc4fa978b9c60bb'​, ​compressedPublicKey​:  ​'0x032a46d3f965d8e0e99785765a9875046712f193b311323f8937a7721007446595'​, ​publicKeyBytes​:  [ ​3​,  ​42​,  ​149​ ] }, ​publicKey​:  '0x042a46d3f965d8e0e99785765a9875046712f193b311323f8937a77210074465953bc0b704a60c  75e3ae70fa47468aea873e35dd44a915486c8bc4fa978b9c60bb'​,  ​address​: ​'0xAd4244FC4add91C7EA2B7911cF54003c53Dfeda1'​ },  ​provider​: ​undefined​ } |

### 4. Save HD Wallet as JSON

Encrypt and save given HD node to a ​**JSON​** document by ​**password​**.

To save the HD Wallet in an encrypted JSON format, you need the Wallet to include the mnemonic phrase. The mnemonic is encrypted in the "​**x-ethers​**" part of the json.



### 5. Load HD Wallet from JSON

Decrypt and load an HD node from a ​**JSON​** document using a ​**password​**.

Use ​**ethers.Wallet.fromEncryptedJson(json, password)**

async​ ​function​ ​decryptWallet​(json, password) {

​return​ ethers.Wallet.fromEncryptedJson(json, password);

}

|  |
| --- |
| (​async​() => {  ​let​ wallet = ethers.Wallet.createRandom();  ​let​ password = ​"p@$$word"​;  ​let​ json = ​await​ saveWalletAsJson(wallet, password);    ​let​ walletDecrypted = ​await​ decryptWallet(json, password);  ​console​.log(walletDecrypted); })(); |

Console output:

|  |
| --- |
| Wallet {  ​signingKey​: SigningKey { ​mnemonic​:  ​'rapid strong noise uncover radio supreme pair grief still twenty blouse pigeon'​,  ​path​: ​"m/2147483692'/2147483708'/2147483648'/0/0"​, ​privateKey​:  '0x87cabd9808e460fdf288473313b440934134ade8fda7edb3dc062e895eed7315'​, ​keyPair​:  KeyPair { ​privateKey​:  '0x87cabd9808e460fdf288473313b440934134ade8fda7edb3dc062e895eed7315'​, ​publicKey​:  '0x0426135748724427f4d3431e7c297fe162768ac6956b599614c73714e2f6d74f69766b15026099 95add37e9b2129bab82d06de84d722c12dd8ed0dfc2ce1322d43'​, ​compressedPublicKey​:  ​'0x0326135748724427f4d3431e7c297fe162768ac6956b599614c73714e2f6d74f69'​, ​publicKeyBytes​:  [ ​3​,  ​38​,  ​79​,  ​105​ ] }, ​publicKey​:  '0x0426135748724427f4d3431e7c297fe162768ac6956b599614c73714e2f6d74f69766b15026099 |

95add37e9b2129bab82d06de84d722c12dd8ed0dfc2ce1322d43'​,

​address​: ​'0x97aCA83E0E3e506e90CeCFEeAD58F0940c993056' },​

​provider​: ​undefined​ }

### 6. Derive Keys from HD Wallet

**Derive keys​** (and their associated addresses) from HD Wallet by given derivation path. Derivation path is ​**m/44'/60'/0'/0**

|  |
| --- |
| function​ ​deriveFiveWalletsFromHdNode​(mnemonic, derivationPath) { ​let​ wallets = [];    ​for​ (​let​ i = ​0​; i < ​5​; i++) {  ​let​ hdNode = ethers.utils.HDNode.fromMnemonic(mnemonic).derivePath(derivationPath + ​"/" ​+ i);  ​console​.log(hdNode);  ​let​ wallet = ​new​ ethers.Wallet(hdNode.privateKey); wallets.push(wallet); }    ​return​ wallets;  } let​ mnemonic = ​'upset fuel enhance depart portion hope core animal innocent will athlete snack'​;  let​ derivationPath = ​"m/44'/60'/0'/0"​;    console​.log(deriveFiveWalletsFromHdNode(mnemonic, derivationPath)); |

Console output:

HDNode {

​keyPair​:

|  |
| --- |
| KeyPair { ​privateKey​:  '0x3ea717d930e238d8471e12786b423d185f80da538a1d12347dcd23e8e6c9aaa8'​, ​publicKey​:    '0x04ca37eef6892576ac42d95161714bfd29c3ecf6d4c539c6a810d458960081bcaf55d580417fb 63a70ad9e5dd966d98d579498b5c9ddba1feda33c8689f35a9059'​, ​compressedPublicKey​:  '0x03ca37eef6892576ac42d95161714bfd29c3ecf6d4c539c6a810d458960081bcaf'​, ​publicKeyBytes​:  [ ​3​,  ​129​,  ​188​,  ​175​ ] }, ​privateKey​:  ​'0x3ea717d930e238d8471e12786b423d185f80da538a1d12347dcd23e8e6c9aaa8'​, ​publicKey​:  ​'0x03ca37eef6892576ac42d95161714bfd29c3ecf6d4c539c6a810d458960081bcaf'​, ​address​: ​'0x909bEE30e27F0fd2aa0A3828247d2D4DE51E260D'​, ​chainCode​:  '0x7eded4b8b8a264f0d9d77d3a645743f5675e74209b39fbb90c77b5568171fec3'​,  ​index​: ​0​,  ​depth​: ​4​, ​mnemonic​:  ​'upset fuel enhance depart portion hope core animal innocent will athlete snack'​,  ​path​: ​"m/2147483692'/2147483708'/2147483648'/0"​ }  HDNode {  ​keyPair​:  KeyPair { ​privateKey​:  '0xe5cf054f346d1734a9476f3d89e1372242f74f5b0b36b0f0315f0bfee9564daf'​, ​publicKey​:  '0x0478ea765484fa3758afb95f70339847acdfa9ea8f8fc56aea7e155d5b4d5be7eb63577604607 afcae31f6632cf5aab39d842d835d818e9c3dd38eda2b6422c327'​, ​compressedPublicKey​:  '0x0378ea765484fa3758afb95f70339847acdfa9ea8f8fc56aea7e155d5b4d5be7eb'​, ​publicKeyBytes​: |

|  |
| --- |
| [ ​3​,  ​231​,  ​235​ ] }, ​privateKey​:  '0xe5cf054f346d1734a9476f3d89e1372242f74f5b0b36b0f0315f0bfee9564daf'​, ​publicKey​:  '0x0378ea765484fa3758afb95f70339847acdfa9ea8f8fc56aea7e155d5b4d5be7eb'​, ​address​: ​'0x5e61371659d93885871E802176aae7f387274e3D'​, ​chainCode​:  '0xc2993df76adf1e3ea1a7adb69162e00b561ba7cd18ba10840d4b16e07a0f4fc7'​,  ​index​: ​1​,  ​depth​: ​4​, ​mnemonic​:  ​'upset fuel enhance depart portion hope core animal innocent will athlete snack'​,  ​path​: ​"m/2147483692'/2147483708'/2147483648'/1"​ }  HDNode {  ​keyPair​:  KeyPair { ​privateKey​:  '0xbf40dcca3cc34e7fd8aaf4aeaa39c71e9f708e497a9a6acd9512e0e33992b890'​, ​publicKey​:  '0x045d5e89a8afe7f240ee33312ba4c8e9d8c49537ca14ad6d4348a9b07c470fa70afb09d44414d 333a5fbc3540f7d30794d12ecc88cba496d73a80d201e57955241'​, ​compressedPublicKey​:  '0x035d5e89a8afe7f240ee33312ba4c8e9d8c49537ca14ad6d4348a9b07c470fa70a'​, ​publicKeyBytes​:  [ ​3​,  ​15​,  ​167​,  ​10​ ] }, ​privateKey​:  '0xbf40dcca3cc34e7fd8aaf4aeaa39c71e9f708e497a9a6acd9512e0e33992b890'​, ​publicKey​:  '0x035d5e89a8afe7f240ee33312ba4c8e9d8c49537ca14ad6d4348a9b07c470fa70a'​, ​address​: ​'0xa223642bC55497a72B5D60bCeDee464d8bFd4fe7'​, ​chainCode​:  '0xdba67395981de8ce6fced1ca8be9c64f34c9dc31d65b8ca9050448d4ea405a20'​, |

|  |
| --- |
| ​index​: ​2​,  ​depth​: ​4​, ​mnemonic​:  ​'upset fuel enhance depart portion hope core animal innocent will athlete snack'​,  ​path​: ​"m/2147483692'/2147483708'/2147483648'/2"​ }  HDNode {  ​keyPair​:  KeyPair { ​privateKey​:  '0x20d814eabcfbeb58cd856d9400388268e6054bc47aca26dee8827862828f41d5'​, ​publicKey​:    '0x04ec5f89b674ae462f55ce5e2fb8ac5bd388366ca87462a57b23aece20f2f01da6e9c5b4cb78b 0fc34db1fee010a00a71562d2778a1966306dfb08ef3622575b27'​, ​compressedPublicKey​:  '0x03ec5f89b674ae462f55ce5e2fb8ac5bd388366ca87462a57b23aece20f2f01da6'​, ​publicKeyBytes​:  [ ​3​,  ​85​,  ​29​, ​166​ ] }, ​privateKey​:  '0x20d814eabcfbeb58cd856d9400388268e6054bc47aca26dee8827862828f41d5'​, ​publicKey​:  '0x03ec5f89b674ae462f55ce5e2fb8ac5bd388366ca87462a57b23aece20f2f01da6'​, ​address​: ​'0x4FCBC9018CB780Ba0962E52361E1055e39CBC63A'​, ​chainCode​:  '0xa2b6b8aaf573024f1576dccabc93e377552dcadcd7bb6279f71831b86042d905'​,  ​index​: ​3​,  ​depth​: ​4​, ​mnemonic​:  ​'upset fuel enhance depart portion hope core animal innocent will athlete snack'​,  ​path​: ​"m/2147483692'/2147483708'/2147483648'/3"​ }  HDNode {  ​keyPair​:  KeyPair { |

|  |
| --- |
| ​privateKey​:  '0x2b3b52543464d37abb3de104efb9715f373bf8887fa83734481dfbce27bf8136'​, ​publicKey​:  '0x049361ab1008f0a3dbdcee67c53e8549f028bfad6583490163ccb15354b6617d664040fcea881 54e1be1efc383bd889bc2a8934e9975f5bff373b81d31a0e915fb'​, ​compressedPublicKey​:  '0x039361ab1008f0a3dbdcee67c53e8549f028bfad6583490163ccb15354b6617d66'​, ​publicKeyBytes​:  [ ​3​,  ​147​, ​8​,  ​125​,  ​102​ ] }, ​privateKey​:  '0x2b3b52543464d37abb3de104efb9715f373bf8887fa83734481dfbce27bf8136'​, ​publicKey​:  '0x039361ab1008f0a3dbdcee67c53e8549f028bfad6583490163ccb15354b6617d66'​, ​address​: ​'0x86279FE676A5691D7e493B9f2B56f6F097cAF97B'​, ​chainCode​:  '0x2bcd06593598437a284de7efe3fe5702d80a009772938a5d92633da9b79e6090'​,  ​index​: ​4​,  ​depth​: ​4​, ​mnemonic​:  ​'upset fuel enhance depart portion hope core animal innocent will athlete snack'​,  ​path​: ​"m/2147483692'/2147483708'/2147483648'/4"​ }  [ Wallet {  ​signingKey​:  SigningKey { ​privateKey​:  '0x3ea717d930e238d8471e12786b423d185f80da538a1d12347dcd23e8e6c9aaa8'​, ​keyPair​:  KeyPair { ​privateKey​:  '0x3ea717d930e238d8471e12786b423d185f80da538a1d12347dcd23e8e6c9aaa8'​, ​publicKey​:  '0x04ca37eef6892576ac42d95161714bfd29c3ecf6d4c539c6a810d458960081bcaf55d580417fb  63a70ad9e5dd966d98d579498b5c9ddba1feda33c8689f35a9059'​, |

|  |
| --- |
| ​compressedPublicKey​:  '0x03ca37eef6892576ac42d95161714bfd29c3ecf6d4c539c6a810d458960081bcaf'​, ​publicKeyBytes​:  [ ​3​,  ​202​,  ​188​,  ​175​ ] }, ​publicKey​:  '0x04ca37eef6892576ac42d95161714bfd29c3ecf6d4c539c6a810d458960081bcaf55d580417fb  63a70ad9e5dd966d98d579498b5c9ddba1feda33c8689f35a9059'​,  ​address​: ​'0x909bEE30e27F0fd2aa0A3828247d2D4DE51E260D'​ },  ​provider​: ​undefined​ },  Wallet {  ​signingKey​:  SigningKey { ​privateKey​:  '0xe5cf054f346d1734a9476f3d89e1372242f74f5b0b36b0f0315f0bfee9564daf'​, ​keyPair​:  KeyPair { ​privateKey​:  '0xe5cf054f346d1734a9476f3d89e1372242f74f5b0b36b0f0315f0bfee9564daf'​, ​publicKey​:  '0x0478ea765484fa3758afb95f70339847acdfa9ea8f8fc56aea7e155d5b4d5be7eb63577604607 afcae31f6632cf5aab39d842d835d818e9c3dd38eda2b6422c327'​, ​compressedPublicKey​:  '0x0378ea765484fa3758afb95f70339847acdfa9ea8f8fc56aea7e155d5b4d5be7eb'​, ​publicKeyBytes​:  [ ​3​,  ​231​,  ​235​ ] }, ​publicKey​:  '0x0478ea765484fa3758afb95f70339847acdfa9ea8f8fc56aea7e155d5b4d5be7eb63577604607 afcae31f6632cf5aab39d842d835d818e9c3dd38eda2b6422c327'​,  ​address​: ​'0x5e61371659d93885871E802176aae7f387274e3D'​ },  ​provider​: ​undefined​ },  Wallet {  ​signingKey​:  SigningKey { |

|  |
| --- |
| ​privateKey​:  '0xbf40dcca3cc34e7fd8aaf4aeaa39c71e9f708e497a9a6acd9512e0e33992b890'​, ​keyPair​:  KeyPair { ​privateKey​:  ​'0xbf40dcca3cc34e7fd8aaf4aeaa39c71e9f708e497a9a6acd9512e0e33992b890'​, ​publicKey​:  '0x045d5e89a8afe7f240ee33312ba4c8e9d8c49537ca14ad6d4348a9b07c470fa70afb09d44414d 333a5fbc3540f7d30794d12ecc88cba496d73a80d201e57955241'​, ​compressedPublicKey​:  '0x035d5e89a8afe7f240ee33312ba4c8e9d8c49537ca14ad6d4348a9b07c470fa70a'​, ​publicKeyBytes​:  [ ​3​,  ​93​,  ​167​,  ​10​ ] }, ​publicKey​:  '0x045d5e89a8afe7f240ee33312ba4c8e9d8c49537ca14ad6d4348a9b07c470fa70afb09d44414d  333a5fbc3540f7d30794d12ecc88cba496d73a80d201e57955241'​,  ​address​: ​'0xa223642bC55497a72B5D60bCeDee464d8bFd4fe7'​ },  ​provider​: ​undefined​ },  Wallet {  ​signingKey​:  SigningKey { ​privateKey​:  '0x20d814eabcfbeb58cd856d9400388268e6054bc47aca26dee8827862828f41d5'​, ​keyPair​:  KeyPair { ​privateKey​:  '0x20d814eabcfbeb58cd856d9400388268e6054bc47aca26dee8827862828f41d5'​, ​publicKey​:  '0x04ec5f89b674ae462f55ce5e2fb8ac5bd388366ca87462a57b23aece20f2f01da6e9c5b4cb78b 0fc34db1fee010a00a71562d2778a1966306dfb08ef3622575b27'​, ​compressedPublicKey​:  '0x03ec5f89b674ae462f55ce5e2fb8ac5bd388366ca87462a57b23aece20f2f01da6'​, ​publicKeyBytes​:  [ ​3​,  ​240​, |
| ​166​ ] }, ​publicKey​:    '0x04ec5f89b674ae462f55ce5e2fb8ac5bd388366ca87462a57b23aece20f2f01da6e9c5b4cb78b  0fc34db1fee010a00a71562d2778a1966306dfb08ef3622575b27'​,  ​address​: ​'0x4FCBC9018CB780Ba0962E52361E1055e39CBC63A'​ },  ​provider​: ​undefined​ },  Wallet {  ​signingKey​:  SigningKey { ​privateKey​:  '0x2b3b52543464d37abb3de104efb9715f373bf8887fa83734481dfbce27bf8136'​, ​keyPair​:  KeyPair { ​privateKey​:  '0x2b3b52543464d37abb3de104efb9715f373bf8887fa83734481dfbce27bf8136'​, ​publicKey​:  '0x049361ab1008f0a3dbdcee67c53e8549f028bfad6583490163ccb15354b6617d664040fcea881 54e1be1efc383bd889bc2a8934e9975f5bff373b81d31a0e915fb'​, ​compressedPublicKey​:  '0x039361ab1008f0a3dbdcee67c53e8549f028bfad6583490163ccb15354b6617d66'​, ​publicKeyBytes​:  [ ​3​,  ​147​,  ​102​ ] }, ​publicKey​:  '0x049361ab1008f0a3dbdcee67c53e8549f028bfad6583490163ccb15354b6617d664040fcea881  54e1be1efc383bd889bc2a8934e9975f5bff373b81d31a0e915fb'​,  ​address​: ​'0x86279FE676A5691D7e493B9f2B56f6F097cAF97B'​ },  ​provider​: ​undefined​ } ] |

### 7. Sign a Transaction

Take the second of the derived wallets and sign a transaction with a given recipient address and ether value.

async​ ​function​ ​signTransaction​(wallet, toAddress, value) {

​let​ transaction = {

​nonce​: ​0​,

​gasLimit​: ​21000​,

​gasPrice​: ethers.utils.bigNumberify(​"2000000000"​),

​to​: toAddress,

​value​: ethers.utils.parseEther(value),

​data​: ​"0x"

};

​return​ wallet.sign(transaction);

}

Ethereum Recipient Address :

0x933b946c4fec43372c5580096408d25b3c7936c5

|  |
| --- |
| let​ mnemonic = ​'upset fuel enhance depart portion hope core animal innocent will athlete snack'​;  let​ derivationPath = ​"m/44'/60'/0'/0"​;  let​ wallets = deriveFiveWalletsFromHDNode(mnemonic, derivationPath); let​ wallet = wallets[​1​];  let​ recipient = ​"0x933b946c4fec43372c5580096408d25b3c7936c5"​; let​ value = ​"1.0"​; |

(​async​() => {

​let​ signedTransaction = ​await​ signTransaction(wallet, recipient, value);

​console​.log(​"Signed Transaction:\n"​ + signedTransaction); })();

Signed Transaction:

0xf86b80847735940082520894933b946c4fec43372c5580096408d25b3c793 6c5880de0b6b3a7640000801ba0de4eb9855d25f65326562730d5e20d4af8a bb48bf36067043d1ed2b1dc09f989a048f06937d9398cb54b898aa32b5d4c97

02f37571e790e44a139aa6e9ba9522fa

**What to Submit?**

Create a ​**ZIP file​** (e.g. ​**your-name-ethereum-hd-wallet-exercise.zip​**) holding your source code for all problems. Submit your ZIP file as ​**homework ​**on the course website.